**Objeto XMLHttpRequest Object**

É o objeto que será utilizado para fazer requisições AJAX.

**Criando um objeto XMLHttpRequest**

**Ex.:** *const obj = new XMLHttpRequest()*

**Criando um Request**

**Esepcificando o tipo de Request**

**obj.open(method, url, async, user, password):** método que especifica o tipo de HTTP request. Onde:

* **Method:** método http que será usado
* **Url:** localização da página ou arquivo requisitado
* **Async:** determina se a requisição será assíncrona ou não.
* **User(opcional):** usuário para ser usado para autenticação
* **Password(opcional):** senha para ser usada para autenticação

**Enviando a requisição**

**obj.send(body):** envia uma requisição. Onde:

* **Body(opcional):** o corpo da requisição, usado, por exemplo, pelo método POST, e pode ser:
  + **Um documento(HTML, Texto, JSON etc.)**
  + **XMLHttpRequestBodyInit, URLSearchParameters, FormData**
  + **Null(valor padrão)**

**Eventos do Objeto XMLHttpRequest**

* **Loadstart:** quando o request começou
* **Load:** quando os dados terminaram de ser transferidos
* **Loadend:** quando a transferência de dados terminar, com sucesso ou não
* **Progress:** evento que ocorre periodicamento enquanto a transferência de dados estiver ocorrendo
* **Error:** quando houver algum erro na transferência de dados
* **Abort:** quando a requisição for abortada
* **Readystatechange:** quando a propriedade ***readyState*** mudar
* **Timeout:** quando o tempo da requisição ultrapassar um tempo de expiração configurado

**Propriedades Próprias dos Eventos**

* **event.lengthComputable(boolean):** indica que o recurso requisitado tem comprimento que pode ser calculado
* **event.loaded:** número indicando o progresso até agora. Se for download, representa o número de bytes que foram baixados; upload representa o números de bytes sendo enviados
* **event.total:** indica o número total de dados sendo transmitidos ou processado.

**Propriedades do objeto XMLHTTPRequest**

**Propriedades relacionadas com a Requisição**

**obj.readyState:** define a situação da requisição atual. Sendo

|  |  |  |
| --- | --- | --- |
| **Valor** | **Stado** | **Descrição** |
| 0 | UNSENT | Objeto XMLHttpRequest criado, mas o método open() ainda não foi chamado |
| 1 | OPENED | Método open() foi chamado |
| 2 | HEADERS\_RECEIVED | Método send() foi chamado e o cabeçalho de resposta foi recebido |
| 3 | LOADING | Recebendo os dados |
| 4 | DONE | Operação terminou |

**Propriedades relacionadas com o Código de Status**

**obj.status:** retorna o código de status HTTP da resposta.

* O valor antes da requisição ser enviada com o método send() é 0
* Em caso de erros, o valor também é 0

**obj.statusText:** retorna a descrição do código de status

* O valor inicial será uma **string vazia** antes do pedido ser enviado, ou seja, antes de chamar o método send()

**Propriedades relacionadas com a Respostas**

**obj.response:** retorna o conteúdo do corpo da resposta, de acordo com o que foi especificado na propriedade **XMLHttpRequest.responseType**

* Recebe null se a requisição ainda estiver sendo feita ou não tiver sucesso ou uma string vazia se o responseType for texto

**obj.responseType:** especifica o tipo de dado contido no corpo da resposta. Valores aceitos:

* **“text”:** formato de exto
* **“ ”:** o mesmo que texto
* **“json”:** recebe uma string em formato JSON
* **“document”:** um documento HTML ou XML
* **“blob”:** objeto contendo dados binários
* **“arraybuffer”:** array contendo dados binários

**obj.responseText:** retorna o texto recebido.

* **XMLHttpRequest.responseType** deve ser configurado para “ ” ou “text” para pode usar esta proprieadade
* **responseText** sempre vai conter o conteúdo de texto requisitado atual, mesmo que a requisição ainda não tenha terminado
* **reponseXML:** recebe os dados em formnato XML

**obj.responseURL:** retorna um URL requisitada

* retorna uma string vazia se a URL estiver nula
* qualquer fragmento da URL será retirado

**Métodos do objeto XMLHTTPRequest**

**Obj.getResponseHeader(nome):** retorna o valor de um header de resposta com o nome especificado.

* É case-insensetive, ou seja, não diferencia letras maiúsculas ou minúsculas
* Retorna null se o header não existir

**Ex.:** obj.getResponseHeader(‘Content-Type’)

**Obj.getAllResponseHeader():** retorna todos os headers de resposta separado em forma de string separada por CRLF(\r\n)

* Retorna null se não tiver response

**Ex.:** mostrando todos os cabeçalhos de resposta



**Lidando com Uploads**

**XMLHttpRequest.upload:** é um objeto para lidar com uploads

* Objeto upload também tem eventos
* Event Listener devem ser adicionados antes do método send()
* Também devem ser adicionados antes do método open()

|  |  |
| --- | --- |
| **Evento** | **Descrição** |
| loadstart | Quando o upload começar |
| progress | Enquanto estiver upando |
| abort | Quando a operação for abortada |
| error | Quando o upload falhar |
| load | Quando o upload terminar |
| timeout | Quando a operação exceder o tempo especificado em **XMLHttpRequest.timeout** |
| loadend | Quando a operação terminar, independentemente do resultado |